APPENDIX B2

```
/*************************************************************
 *
 * U.S. Patent Pending. Copyright 2000 Yahoo! Inc.,
 * 3420 Central Expressway, Santa Clara, California U.S.A.
 *
 * ALL RIGHTS RESERVED.
 *
 * This computer program is protected by copyright law and
 * international treaties. Unauthorized reproduction or
 * distribution of this program, or any portion of it, may
 * result in severe civil and criminal penalties, and will
 * be prosecuted to the maximum extent possible under the law.
 *
 *************************************************************/


/*************************************************************
 * String Utilities
 *************************************************************/

function rjs_startsWith(full, sub)
{
    var fullLower = full.toLowerCase();
    var subLower = sub.toLowerCase();
    var index = fullLower.indexOf(subLower);
    return index ? false : true;
}

function rjs_endsExactlyWith(full, sub)
{
    var offset = full.length - sub.length;
    if (offset < 0) return false;

    var index = full.indexOf(sub, offset);
    return (index==offset) ? true : false;
}

/*************************************************************
 * Debug utilities
 *************************************************************/
function rjs_viewObj(obj)
{
    for (i in obj) alert("rjs_viewObj() : " + i + "=" + obj[i]);
}

/*************************************************************
 * Is the string at the end of left-hand side of '='
 *************************************************************/
function rjs_isEndOfLHS(sub)
{
    if (rjs_AssignmentState != "lhs") return false;

    if (rjs_endsExactlyWith(rjs_Tokens.str() , sub) )
        return true;
    else
```

```
        return false;
}

/************************************************************
 * Find the next token (from 'startPos' to the end) equals
 * to 'str' & return the index
 ************************************************************/
function rjs_findNext(startPos, str)
{
    for (var i=startPos; i<rjs_Tokens.length; ++i)
        if (rjs_Tokens[i] == str) return i;

    return -1;    // not found
}

/************************************************************
 * Find the last token (between 'head_pos' & 'tail_pos') equals
 * to 'str' & return the index
 ************************************************************/
function rjs_findLast(head_pos, tail_pos, str)
{
    for (var i=tail_pos; i >= head_pos; --i)
        if (rjs_Tokens[i] == str) return i;

    return -1;    // not found
}

/*****************************************************
 * Begin inserting "rmi_xlateURL(*)"
 *****************************************************/
function rjs_xUrlBegin(str)
{
    rjs_XUrl_setLocationTail = "";

    // Is top or parent in the chain?

    var top_pos = rjs_findNext(rjs_Index_id, "top");
    var parent_pos = rjs_findNext(rjs_Index_id, "parent");

    if (top_pos != -1 || parent_pos != -1)
    {
        var split_pos = rjs_Index_id;

        if (top_pos == -1)
            split_pos = parent_pos;
        else if (parent_pos == -1)
            split_pos = top_pos;
        else
            split_pos = top_pos;   // use 'top' if both are found

        var head = rjs_Tokens.section(rjs_Index_id, split_pos);
        var rest = rjs_Tokens.section(split_pos+2);              // skip
.
        var override = "rmi_setLocation(\"" + head + "\", \"" + rest +
"\", rmi_xlateURL(";

        // Get "a.b.c" from "a.b.c.location"
```

```
        var loc_pos = rjs_findNext(rjs_Index_id, "location");
        var win = rjs_Tokens.section(rjs_Index_id, loc_pos-2);

        rjs_XUrl_setLocationTail = "), " + win + ")";
        str = override;
        rjs_Tokens.null_section(rjs_Index_id);     // null
a.top.b.location.href
    }

    rjs_XUrl_on = true;
    return str;
}

/*****************************************************
 * Finish inserting "rmi_xlateURL(*)"
 *****************************************************/
function rjs_xUrlEnd(str)
{
    rjs_XUrl_on = false;

    if (rjs_XUrl_setLocationTail != "")
        return rjs_XUrl_setLocationTail;
    else
        return str;
}

/*****************************************************
 * Begin inserting "rmi_setCookie(*)"
 *****************************************************/
function rjs_xCookieBegin(str)
{
    rjs_XCookie_on = true;

    // remove "o.document.cookie" in  "o.document.cookie ="
    var cur = rjs_Tokens.length-1;
    rjs_Tokens.null_section(rjs_Index_id, cur);

    return str;
}

/*****************************************************
 * Finish inserting "rmi_setCookie(*)"
 *****************************************************/
function rjs_xCookieEnd(str)
{
    rjs_XCookie_on = false;
    return str;
}

/*****************************************************
 * Begin inserting "rmi_xlateURL(*)" for "*.action="
 *****************************************************/
function rjs_xActionBegin(str)
{
    rjs_XAction_on = true;
    return str;
}
```

```
/*****************************************************
 * Finish inserting "rmi_xlateURL(*)" for "*.action="
 *****************************************************/
function rjs_xActionEnd(str)
{
    rjs_XAction_on = false;
    return str;
}

/*****************************************************
 * Begin inserting "rmi_xlate(*)"
 *****************************************************/
function rjs_xInnerHtmlBegin(str)
{
    rjs_XInnerHtml_on = true;
    return str;
}

/*****************************************************
 * Finish inserting "rmi_xlate(*)"
 *****************************************************/
function rjs_xInnerHtmlEnd(str)
{
    rjs_XInnerHtml_on = false;
    return str;
}


/*****************************************************
 * Translate "document.layers"
 *****************************************************/
function rjs_xLayers(str)
{
    if (rjs_LayerState != "doc") return str;

    var pre = rjs_Tokens.length-3;
    var cur = rjs_Tokens.length-1;
    if (pre < 0 || cur < 0) return str;

    if (rjs_Tokens[pre] == "document" && rjs_Tokens[cur] == "layers")
    {
        rjs_Tokens[pre] = "document.layers[\'rmilayer\'].document";
        rjs_LayerState = "";
    }

    return str;
}


/*************************************************************
 * Save current token position into a global variable
 * (e.g. rjs_Index_id)
 *************************************************************/
var rjs_Index_id = 0;

function rjs_saveIndexFor(type)
```

```
{
    var code = "rjs_Index_" + type + " = rjs_Tokens.length - 1";
    eval(code);
}

/***************************************************
 * Increment top elements for ALL 'nesting' arrays
 ***************************************************/
function rjs_incTopForNesting()
{
    rjs_incTop(rjs_XUrl_nesting);
    rjs_incTop(rjs_XCookie_nesting);
    rjs_incTop(rjs_XAction_nesting);
    rjs_incTop(rjs_XInnerHtml_nesting);
}

/***************************************************
 * Decrement top elements for ALL 'nesting' arrays
 ***************************************************/
function rjs_decTopForNesting()
{
    rjs_decTop(rjs_XUrl_nesting);
    rjs_decTop(rjs_XCookie_nesting);
    rjs_decTop(rjs_XAction_nesting);
    rjs_decTop(rjs_XInnerHtml_nesting);
}

/***************************************************
 * Increment the top element of an array
 ***************************************************/
function rjs_incTop(array)
{
    var cur = array.length - 1;
    if (cur < 0) cur = 0;

    return ++array[cur];
}

/***************************************************
 * Decrement the top element of an array
 ***************************************************/
function rjs_decTop(array)
{
    var cur = array.length - 1;
    if (cur < 0)  cur = 0;
    return --array[cur];
}

/***************************************************
 * Return (NOT pop!) the top element of an array
 ***************************************************/
function rjs_retTop(array)
{
    var cur = array.length - 1;
    if (cur < 0)  cur = 0;
    return array[cur];
}
```

```
/*****************************************************
 * Save head & tail positions for a chain
 * (e.g. for "a.b.c.d", positions of a & d are saved)
 *
 * Previous 'identifier' position is saved as a head position
 * Relative offset from current position is saved as a tail position
 *****************************************************/
function rjs_saveChain(rel, head, tail)
{
    var cur = rjs_Tokens.length - 1;
    var pre = cur + rel;

    if (pre < 0 || cur < 0) return false;

    head.push(rjs_Index_id);
    tail.push(pre);


    return true;
}

/*********************************************
 * Save *.open() attributes
 *
 * Trigger State: *.open(
 *********************************************/
function rjs_saveOpen()
{
    var cur = rjs_Tokens.length - 1;
    var pre = cur - 1;
    if (pre < 0 || cur < 0) return false;

    if (rjs_Tokens[pre] == "open")
    {
        // e.g. save positions for a & c for "a.b.c.open("
        rjs_saveChain( -3, rjs_Open_head, rjs_Open_tail);

        rjs_OpenFunc_pos.push(pre);       // e.g. save position for
"open"

    }

    return true;
}

/*********************************************
 * Translate open(*) or *.open(*)
 *
 * Trigger State: * . open (*
 *********************************************/
function rjs_xOpen()
{

    if (rjs_OpenFunc_pos.length == 0) return false;

    var func_pos = rjs_OpenFunc_pos.pop();
```

```
    if (rjs_Tokens[func_pos-1] == ".")
    {
        var head_pos = rjs_Open_head.pop();
        var tail_pos = rjs_Open_tail.pop();

        rjs_Tokens[func_pos] = "rmi_winobj_open";
        var arg0 = rjs_Tokens.section(head_pos, tail_pos);
        rjs_Tokens[func_pos+2] = arg0 + ", " + rjs_Tokens[func_pos+2];
        rjs_Tokens.null_section(head_pos, tail_pos + 1);
    }
    else
        rjs_Tokens[func_pos] = "rmi_window_open";

}

/*******************************************
 * Save *.write() & *.writeln() attributes
 *
 * Trigger State: .write( or .writeln(
 *******************************************/
function rjs_saveWrite()
{
    var cur = rjs_Tokens.length - 1;
    var pre0 = cur - 1;
    var pre1 = cur - 2;
    if (pre0 < 0 || pre1 < 0 || cur < 0) return false;

    if (rjs_Tokens[pre1] == ".")
        if (rjs_Tokens[pre0] == "write" || rjs_Tokens[pre0] ==
"writeln")
        {
            // e.g. save positions for a & c for "a.b.c.write("
            rjs_saveChain( -3, rjs_Write_head, rjs_Write_tail);

            rjs_WriteFunc_pos.push(pre0);       // e.g. save position
for "write" or "writeln"
        }

    return true;
}

/*******************************************
 * Translate  *.write(*) or *.writeln(*)
 *
 * Trigger State: .write( or .writeln(
 *******************************************/
function rjs_xWrite()
{
    if (rjs_WriteFunc_pos.length == 0) return false;

    var func_pos = rjs_WriteFunc_pos.pop();

    if (rjs_Tokens[func_pos-1] == ".")
    {
        var head_pos = rjs_Write_head.pop();
        var tail_pos = rjs_Write_tail.pop();
```

```
        rjs_Tokens[func_pos] = "rmi_" + rjs_Tokens[func_pos];    // xlate
write or writeln
        var arg0 = rjs_Tokens.section(head_pos, tail_pos);
        rjs_Tokens[func_pos+2] = arg0 + ", " + rjs_Tokens[func_pos+2];
        rjs_Tokens.null_section(head_pos, tail_pos + 1);
    }
}

/*********************************************
 * Save *.location.replace() attributes
 *
 * Trigger State: *.replace(
 *********************************************/
function rjs_saveReplace()
{
    var cur = rjs_Tokens.length - 1;
    var pre0 = cur - 1;
    var pre1 = cur - 3;
    if (pre0 < 0 || pre1 < 0 || cur < 0) return false;

    if (rjs_Tokens[pre1] == "location" && rjs_Tokens[pre0] == "replace")
    {
        // e.g. save positions for a & c for "a.b.c.location.replace("
        rjs_saveChain(-5, rjs_Replace_head, rjs_Replace_tail);

        rjs_ReplaceFunc_pos.push(pre0);      // e.g. save position for
"replace"
    }

    return true;
}

/*********************************************
 * Translate location.replace(*) or *.location.replace(*)
 *
 * Trigger State: * . replace (*
 *********************************************/
function rjs_xReplace()
{
    if (rjs_ReplaceFunc_pos.length == 0) return false;

    var func_pos = rjs_ReplaceFunc_pos.pop();
    var head_pos = rjs_Replace_head.pop();
    var tail_pos = rjs_Replace_tail.pop();

    if (rjs_Tokens[func_pos-3] == ".")
    {
        // Handle the argument IF top or parent is in the chain

        var top_pos = rjs_findLast(head_pos, tail_pos, "top");
        var parent_pos = rjs_findLast(head_pos, tail_pos, "parent");

        var arg0 = "";

        if (top_pos != -1 || parent_pos != -1)
        {
```

```
            var split_pos = head_pos;

            if (top_pos == -1)
                split_pos = parent_pos;
            else if (parent_pos == -1)
                split_pos = top_pos;
            else
                split_pos = top_pos;    // use 'top' if both are found

            var win = rjs_Tokens.section(head_pos, split_pos);
            var rest = rjs_Tokens.section(split_pos+1, tail_pos);
            arg0 = "rmi_getTop(" + win + ")" + rest;
        }
        else
            arg0 = rjs_Tokens.section(head_pos, tail_pos);

        rjs_Tokens[func_pos] = "rmi_replace";
        rjs_Tokens[func_pos+2] = arg0 + ", " + rjs_Tokens[func_pos+2];
        rjs_Tokens.null_section(head_pos, tail_pos + 3);    // remove
chain.location.
    }
    else
    {
        rjs_Tokens[func_pos+2] = "self, " + rjs_Tokens[func_pos+2];
        rjs_Tokens[func_pos] = "rmi_replace";
        rjs_Tokens.null_section(head_pos, tail_pos + 3);    // remove
chain.location.
    }
}

/*******************************************
 * Save attributes for document.domain or *.document.domain
 *
 * Trigger State: *document.domain
 *******************************************/
function rjs_saveDomain()
{
    if (rjs_endsExactlyWith(rjs_Tokens.str() , "document.domain") )
    {
        // e.g. save positions for a & domain for
"a.b.c.document.domain("
        rjs_saveChain(0, rjs_Domain_head, rjs_Domain_tail);
    }

    return true;
}

/*******************************************
 * Pop attributes for document.domain or *.document.domain
 * for each assignment expression
 *
 * Trigger state: *document.domain in LHS
 *******************************************/
function rjs_popDomain()
{
    rjs_Domain_head.pop();
    rjs_Domain_tail.pop();
```

```
}

/*********************************************
 * Translate document.domain or *.document.domain
 *
 * Trigger State: end of statement
 *********************************************/
function rjs_xDomain()
{
    for (var i=0; i<rjs_Domain_head.length; ++i)
    {
        rjs_Tokens.null_section(rjs_Domain_head[i], rjs_Domain_tail[i]);
        rjs_Tokens[rjs_Domain_head[i]] = "rmi_getOriginalDomain()";
    }
}

/*********************************************
 * Save attributes for location.* or *.location.*
 *
 * Trigger State: *location.* or *location
 *********************************************/
function rjs_saveLocation()
{
    var cur = rjs_Tokens.length - 1;
    var pre0 = cur - 1;
    var pre1 = cur - 2;
    if (pre0 < 0 || pre1 < 0 || cur < 0) return false;

    var str = rjs_Tokens.str();
    var peek = JSC$parser_peek_token_token;

    // if (rjs_Tokens[pre1] == "location" && rjs_Tokens[pre0] == ".")

    if (rjs_endsExactlyWith(str, "location.href")
        || rjs_endsExactlyWith(str, "location.host")
        || rjs_endsExactlyWith(str, "location.hostname")
        || rjs_endsExactlyWith(str, "location.pathname")
        || rjs_endsExactlyWith(str, "location.port")
        || rjs_endsExactlyWith(str, "location.search")
       )
    {
        // e.g. save positions for a & href for "a.b.c.location.href"
        rjs_saveChain(0, rjs_Location_head, rjs_Location_tail);
    }
    else if (rjs_Tokens[cur] == "location" && peek != ".")
    {
        // e.g. save positions for a & location for "a.b.c.location"
        rjs_saveChain(0, rjs_Location_head, rjs_Location_tail);
    }

    return true;
}

/*********************************************
 * Save attributes for standalone 'location'
 *
 * Trigger State: location
```

```
 * (no '.location' or 'location.')
 ********************************************/
function rjs_saveStandaloneLocation()
{
    var cur = rjs_Tokens.length - 1;
    var pre = cur - 1;
    if (pre < 0 || cur < 0) return false;

    if (rjs_Tokens[rjs_Index_id] == "location" && rjs_Index_id == cur &&
rjs_Tokens[pre] != ".")
        rjs_saveChain(0, rjs_Location_head, rjs_Location_tail);
}

/*********************************************
 * Pop attributes for location.* or *.location.*
 * for each assignment expression
 *
 * Trigger state: *location.* in LHS
 *********************************************/
function rjs_popLocation()
{
    rjs_Location_head.pop();
    rjs_Location_tail.pop();
}

/*********************************************
 * Translate *.location, location.*, *.location.*
 *
 * Trigger State: end of statement
 *********************************************/
function rjs_xLocation()
{

    for (var i=0; i<rjs_Location_head.length; ++i)
    {
        var head_pos = rjs_Location_head[i];
        var tail_pos = rjs_Location_tail[i] - 2;
        var arg0 = rjs_Tokens.section(head_pos, tail_pos);

        var prop = rjs_Tokens[rjs_Location_tail[i]];

        if (prop == "location" && arg0 != "")    // from *.location
        {
            arg0 = arg0 + ".location";
            prop = "";
        }
        else if (prop == "location")             // from location
        {
            arg0 = "location";
            prop = "";
        }

        rjs_Tokens.null_section(rjs_Location_head[i],
rjs_Location_tail[i]);
        rjs_Tokens[rjs_Location_head[i]] = "rmi_getOriginal(" + arg0 +
", \"" + prop + "\")";
    }
```

```
    rjs_Location_head.reset();
    rjs_Location_tail.reset();
}

/*********************************************
 * Save attributes for *document.cookie*
 *
 * Trigger State: *document.cookie*
 *********************************************/
function rjs_saveCookie()
{
    var str = rjs_Tokens.str();

    if (rjs_endsExactlyWith(str, "document.cookie"))
    {
        // e.g. save positions for a & cookie for
"a.b.c.document.cookie"
        rjs_saveChain(0, rjs_Cookie_head, rjs_Cookie_tail);
    }

    return true;
}

/*********************************************
 * Pop attributes for *document.cookie*
 * for each assignment expression
 *
 * Trigger state: *document.cookie* in LHS
 *********************************************/
function rjs_popCookie()
{
    rjs_Cookie_head.pop();
    rjs_Cookie_tail.pop();
}

/*********************************************
 * Translate *document.cookie*
 *
 * Trigger State: end of statement
 *********************************************/
function rjs_xCookie()
{
    for (var i=0; i<rjs_Cookie_head.length; ++i)
    {
        var head_pos = rjs_Cookie_head[i];
        var tail_pos = rjs_Cookie_tail[i];
        var arg0 = rjs_Tokens.section(head_pos, tail_pos);

        rjs_Tokens.null_section(rjs_Cookie_head[i], rjs_Cookie_tail[i]);
        rjs_Tokens[rjs_Cookie_head[i]] = "rmi_getCookie(" + arg0 + ")";
    }

    rjs_Cookie_head.reset();
    rjs_Cookie_tail.reset();
}
```

```
/*********************************************
 * Save attributes for *.frames[*].*
 *
 * Trigger State: *.frames[
 *********************************************/
function rjs_saveFrames()
{
    var cur = rjs_Tokens.length - 1;
    var pre = cur - 1;
    if (pre < 0 || pre-1 < 0 || cur < 0) return false;

    if (rjs_Tokens[pre] == "frames" && rjs_Tokens[pre-1] == ".")
    {
        // e.g. save positions for a & c for "a.b.c.frames["
        rjs_saveChain( -3, rjs_Frames_head, rjs_Frames_tail);

        rjs_FramesObj_pos.push(pre);        // e.g. save position for
"frames"
    }

    return true;
}

/*********************************************
 * Translate *.frames[*].*
 *
 * Trigger State: *.frames[
 *********************************************/
function rjs_xFrames()
{
    if (rjs_FramesObj_pos.length == 0) return false;

    var obj_pos = rjs_FramesObj_pos.pop();    // "frames" position

    if (rjs_Tokens[obj_pos-1] == ".")
    {
        var head_pos = rjs_Frames_head.pop();
        var tail_pos = rjs_Frames_tail.pop();

        var left_pos = obj_pos+1;                          // left bracket
position
        var right_pos = rjs_findNext(left_pos, "]");

        if (right_pos != -1)
        {
            rjs_Tokens[obj_pos] = "rmi_getFrame";
            var arg0 = rjs_Tokens.section(head_pos, tail_pos);

            rjs_Tokens[left_pos] = "(" + arg0;

            if (right_pos - left_pos > 1)
                rjs_Tokens[left_pos] += ", ";
            else
                rjs_Tokens[left_pos] += ", 0";    // frames[]

            rjs_Tokens[right_pos] = ")";
```

```
            rjs_Tokens.null_section(head_pos, tail_pos + 1);
        }
    }
}

/*********************************************
 * Translate JavaScript string
 *********************************************/
function rmi_xjs(str)
{
    rjs_Error = false;          // reset

    JSC$generate_debug_info = false;
    JSC$warn_missing_semicolon = false;
    JSC$verbose = false;
    JSC$optimize_constant_folding = false

    var sStr = new JSC$StreamString(str);

    rjs_Stmts.reset();
    JSC$parser_reset ();

    JSC$parser_parse(sStr);

    rjs_debug("OLD:" + str + "\n" + "NEW:" + rjs_Stmts.str() );

    if (rjs_Error)
        return str;
    else
        return rjs_Stmts.str();
}
```